Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по лабораторной работе №3

Тема работы: Массивы

Выполнил

Проверил: Фадеева Е.П.

Минск 2021

содержание

[1 Постановка задачи 3](#_Toc88248422)

[2 Методика решения 4](#_Toc88248423)

[3 Текстовый алгоритм решения задачи 5](#_Toc88248424)

[4 Структура данных 8](#_Toc88248425)

[5 Схема алгоритма решения задачи по ГОСТ 19.701-90 9](#_Toc88248426)

[6 Результаты расчетов 14](#_Toc88248427)

[Приложение А 15](#_Toc88248428)

[Приложение Б 17](#_Toc88248429)

[Приложение В 19](#_Toc88248430)

[Приложение Г 21](#_Toc88248431)

[Приложение Д 23](#_Toc88248432)

[Приложение Е 26](#_Toc88248433)

[Приложение Ж 28](#_Toc88248434)

[Приложение З 30](#_Toc88248435)

[Приложение И 32](#_Toc88248436)

# Постановка задачи

Постановка задачи 1

Дана последовательность целых чисел .

Определить: сколько чисел входит в эту последовательность по одному разу (и какие);

Вывести на печать результаты расчётов:

- различные элементы, - количество различных чисел.

Постановка задачи 2

Дана последовательность целых чисел .

Определить: максимальное из чисел, входящих в последовательность более одного раза;

Вывести на печать результаты расчётов:

- наибольший элемент.

# Методика решения

Решение задачи по постановке 1

Чтобы найти не повторяющиеся элементы в массиве, необходимо сравнить каждый его элемент с каждым его элементом. Это можно сделать при помощи двух циклов. Далее, если элемент не повторяется, он записывается во второй, изначально пустой, массив, где ему присваевается номер. Затем этот элемент выводится на экран, а количество уникальных элементов увеличивается на один. В конце, если количество уникальных элементов больше нуля, программа выводит его.

Решение задачи по постановке 2

Чтобы найти повторяющиеся элементы в массиве, необходимо сравнить каждый его элемент с каждым его элементом. Это можно сделать при помощи двух циклов. Далее, если элемент повторяется, он записывается во второй, изначально пустой, массив, где ему присваевается номер. Количество повторяющихся элементов увеличивается на один.

Как только получен массив повторяющихся элементов, необходимо найти наибольший из элементов. Для этого используется цикл, сравнивающий каждое следующий элемент с предыдущим. Если следующий элемент больше, значит, он является максимальным. По окончании цикла максимальный элемент выводится на экран, при условии, что количество повторяющихся элементов больше нуля.

# Текстовый алгоритм решения задачи

Таблица – Алгоритм решения задачи по постановке 1

|  |  |
| --- | --- |
| Номер  шага | Назначение шага |
|  | Ввод Limit |
|  | I:= 1 |
|  | Начало цикла А1. Проверка выполнения условия (I <= Limit). Если условие истинно, перейти к шагу 4, иначе – к шагу 7 |
|  | Ввод Arr[I] |
|  | I:= I + 1 |
|  | Конец цикла А1. Идти к шагу 3 |
|  | K:= 1 |
|  | Quantity:= 0 |
|  | I:= 1 |
|  | Начало цикла А2. Проверка выполнения условия (I <= Limit). Если условие истинно, перейти к шагу 11, иначе – к шагу 25 |
|  | Unique:= True |
|  | J:= 1 |
|  | Начало цикла А3. Проверка выполнения условия (J <= Limit). Если условие истинно, перейти к шагу 14, иначе – к шагу 18 |
|  | Проверка выполнения условия (Arr1[I] = Arr1[J]) and (J <> I). Если условие истинно, перейти к шагу 15, иначе – к шагу 16 |
|  | Unique:= False |
|  | J:= J + 1 |
|  | Конец цикла А3. Идти к шагу 13 |
|  | Проверка выполнения условия (Unique = True). Если условие истинно, перейти к шагу 19, иначе – к шагу 23 |
|  | Arr2[K]:= Arr1[I] |
|  | Вывод Arr2[K] |
|  | Quantity:= Quantity + 1 |
|  | K:= K + 1 |
|  | I:= I + 1 |
|  | Конец цикла А2. Идти к шагу 10 |
|  | Проверка выполнения условия (Quantity > 0). Если условие истинно, перейти к шагу 26, иначе – к шагу 28 |
|  | Вывод Quantity |
|  | Идти к шагу 29 |
|  | Вывод сообщения о том, что элементов нет |
|  | Останов. |

Таблица 2 – Алгоритм решения задачи по постановке 2

|  |  |
| --- | --- |
| Номер  шага | Назначение шага |
|  | Ввод Limit |
|  | I:= 1 |
|  | Начало цикла А1. Проверка выполнения условия (I <= Limit). Если условие истинно, перейти к шагу 4, иначе – к шагу 7 |
|  | Ввод Arr[I] |
|  | I:= I + 1 |
|  | Конец цикла А1. Идти к шагу 3 |
|  | K:= 1 |
|  | Quantity:= 0 |
|  | I:= 1 |
|  | Начало цикла А2. Проверка выполнения условия (I <= Limit). Если условие истинно, перейти к шагу 11, иначе – к шагу 24 |
|  | Duplicate:= False |
|  | J:= 1 |
|  | Начало цикла А3. Проверка выполнения условия (J <= Limit). Если условие истинно, перейти к шагу 14, иначе – к шагу 18 |
|  | Проверка выполнения условия (Arr1[I] = Arr1[J]) and (J <> I). Если условие истинно, перейти к шагу 15, иначе – к шагу 16 |
|  | Duplicate:= True |
|  | J:= J + 1 |
|  | Конец цикла А3. Идти к шагу 13 |
|  | Проверка выполнения условия (Duplicate = True). Если условие истинно, перейти к шагу 19, иначе – к шагу 22 |
|  | Arr2[K]:= Arr1[I] |
|  | K:= K + 1 |
|  | Quantity:= Quantity + 1 |
|  | I:= I + 1 |
|  | Конец цикла А2. Идти к шагу 10 |
|  | Max:= Arr2[1] |
|  | K:= 2 |
|  | Начало цикла А4. Проверка выполнения условия (K <= Limit). Если условие истинно, перейти к шагу 27, иначе – к шагу 31 |
|  | Проверка выполнения условия (Arr2[K]>Max). Если условие истинно, перейти к шагу 28, иначе – к шагу 29 |
|  | Max:=Arr2[K] |
|  | K:= K + 1 |
|  | Конец цикла А4. Идти к шагу 26 |
|  | Проверка выполнения условия (Quantity > 0). Если условие истинно, перейти к шагу 32, иначе – к шагу 34 |
|  | Вывод Max |

Продолжение таблицы 2

|  |  |
| --- | --- |
| Номер  шага | Назначение шага |
|  | Идти к шагу 35 |
|  | Вывод сообщения о том, что элементов нет |
|  | Останов. |

# Структура данных

Таблица 3 – Данные для решения задачи по постановке 1

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| Arr1 | Array of Integer | Массив, который вводит пользователь |
| Arr2 | Array of Integer | Результат преобразования первоначального массива |
| I, J, K | Integer | Параметры циклов, позиции элементов в массиве |
| Quantity | Integer | Количество элементов без повторений |
| Limit | Integer | Количество элементов массива, которое будет запрошено у пользователя при вводе |
| Unique | Boolean | Иммет ли данный элемент повторения |

Таблица 4 – Данные для решения задачи по постановке 2

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| Arr1 | Integer | Массив, который вводит пользователь |
| Arr2 | Integer | Результат преобразования первоначального массива |
| I, J, K | Integer | Параметры циклов, позиции элементов в массиве |
| Quantity | Integer | Количество элементов с повторениями |
| Limit | Integer | Количество элементов массива, которое будет запрошено у пользователя при вводе |
| Max | Integer | Максимальный из дубликатов |
| Duplicate | Boolean | Иммет ли данный элемент повторения |

# Схема алгоритма решения задачи по ГОСТ 19.701-90

|  |
| --- |
| **По постановке 1** |
|  |
| Рисунок 1 – Схема алгоритма решения задачи по ГОСТ 19.701-90 |

|  |
| --- |
|  |
| Рисунок 2 – Схема алгоритма решения задачи по ГОСТ 19.701-90 |

|  |
| --- |
| **По постановке 2** |
|  |
| Рисунок 3 – Схема алгоритма решения задачи по ГОСТ 19.701-90 |

|  |
| --- |
|  |
| Рисунок 4 – Схема алгоритма решения задачи по ГОСТ 19.701-90 |

# Результаты расчетов

Вследствие результатов программы на экран выводятся следующие результаты расчетов:

По постановке 1:

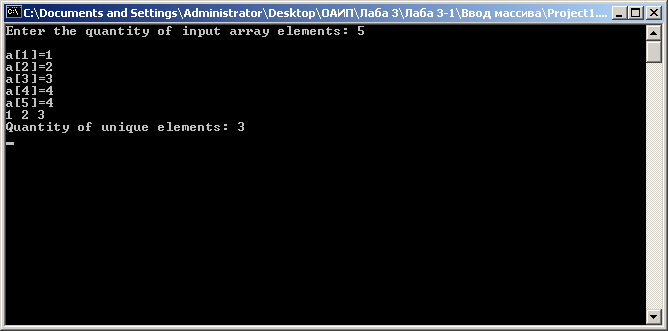


Рисунок 5 – Результаты расчетов

По постановке 2:

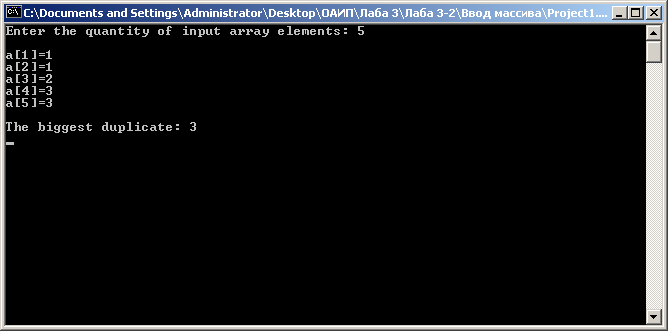


Рисунок 6 – Результаты расчетов

Приложение А

(обязательное)

Исходный код программы (по постановке 1)

Массив заполняется путем ввода данных с клавиатуры

Program Lab3Part1Read;

{Find elements of array without duplicates and their quantity}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare vars

Var

Arr1, Arr2: Array[1..1000] of Integer;

I, J, K, Quantity, Error, Limit: Integer;

Unique: Boolean;

Input: String;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Error - operator for checking input

//Limit - number of elements in array

//Quantity - second array's size

//Unique - if yes, then element has no duplicates

//Input - input strings

Begin

//Checking for the correct input

Repeat

Write('Enter the quantity of input array elements: ');

ReadLn(Input);

Val(Input, Limit, Error);

If (Limit <= 0) or (Error <> 0) then

WriteLn('Invalid input. Enter another number.');

Until (Limit > 0) and (Error = 0);

WriteLn;

//Enter every element of array

For I:= 1 to Limit do

Begin

//Checking for the correct input

Repeat

Write('a[', I, ']=');

ReadLn(Input);

Val(Input, Arr1[I], Error);

If Error <> 0 then

WriteLn('Invalid input. Enter another number.');

Until Error = 0;

End;

K:= 1;

//Prevent from random false number

Quantity:= 0;

//For the every I

For I:= 1 to Limit do

Begin

Unique:= True;

//For the every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Unique:= False;

//Creating and displaying the new array of unique elements

If Unique then

Begin

Arr2[K]:= Arr1[I];

Write(Arr2[K], ' ');

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

WriteLn;

//Displaying the quantity of elements in the new array

If Quantity > 0 then

WriteLn('Quantity of unique elements: ', Quantity)

Else

WriteLn('There is no unique elements in array');

ReadLn;

End.

Приложение Б

(обязательное)

Исходный код программы (по постановке 1)

Массив заполняется с использованием генератора равномерно-распределенных чисел

Program Lab3Part1Generator;

{Find elements of array without duplicates and their quantity}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare vars

Var

Arr1, Arr2: Array[1..1000] of Integer;

I, J, K, Quantity, Error, Limit: Integer;

Unique: Boolean;

Input: String;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Error - operator for checking input

//Limit - number of elements in array

//Quantity - second array's size

//Unique - if yes, then element has no duplicates

//Input - input strings

Begin

//Checking for the correct input

Repeat

Write('Enter the quantity of input array elements: ');

ReadLn(Input);

Val(Input, Limit, Error);

If (Limit <= 0) or (Error <> 0) then

WriteLn('Invalid input. Enter another number.');

Until (Limit > 0) and (Error = 0);

WriteLn;

//Generate and display every element of array

Randomize;

For I:= 1 to Limit do

Begin

Arr1[I]:=Round(Random(1000)/100 - 5);

Write(Arr1[I], ' ');

End;

WriteLn;

K:= 1;

//Prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Unique:= True;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Unique:= False;

//Creating and displaying the new array of unique elements

If Unique then

Begin

Arr2[K]:= Arr1[I];

Write(Arr2[K], ' ');

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

WriteLn;

//Displaying the quantity of elements in the new array

If Quantity > 0 then

WriteLn('Quantity of unique elements: ', Quantity)

Else

WriteLn('There is no unique elements in array');

ReadLn;

End.

Приложение В

(обязательное)

Исходный код программы (по постановке 1)

Массив заполняется с использованием типизированной константы-массив

Program Lab3Part1Const;

{Find elements of array without duplicates and their quantity}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare consts

Const

Limit = 10;

Arr1: Array[1..Limit] of Integer = (1, 1, 3, 4, 5, 6, 7, 8, 9, 9);

//Arr1 - array with duplicates

//Limit - number of elements in array

//Declare vars

Var

Arr2: Array[1..Limit] of Integer;

I, J, K, Quantity: Integer;

Unique: Boolean;

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - second array's size

//Unique - if yes, then element has no duplicates

Begin

//Display every element of array

For I:= 1 to Limit do

Write(Arr1[I], ' ');

WriteLn;

//Checking for the duplicates

K:= 1;

//Prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Unique:= True;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Unique:= False;

//Creating and displaying the new array of unique elements

If Unique then

Begin

Arr2[K]:= Arr1[I];

Write(Arr2[K], ' ');

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

WriteLn;

//Displaying the quantity of elements in the new array

If Quantity > 0 then

WriteLn('Quantity of unique elements: ', Quantity)

Else

WriteLn('There is no unique elements in array');

ReadLn;

End.

Приложение Г

(обязательное)

Исходный код программы (по постановке 1)

Массив заполняется с использованием типизированной переменной-массив

Program Lab3Part1Var;

{Find elements of array without duplicates and their quantity}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare consts

Const

Limit = 10;

//Limit - number of elements in array

//Declare vars

Var

Arr1: Array[1..Limit] of Integer = (1, 1, 3, 4, 5, 6, 7, 8, 9, 9);

Arr2: Array[1..Limit] of Integer;

I, J, K, Quantity: Integer;

Unique: Boolean;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - second array's size

//Unique - if yes, then element has no duplicates

Begin

//Display every element of array

For I:= 1 to Limit do

Write(Arr1[I], ' ');

WriteLn;

K:= 1;

//Prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Unique:= True;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Unique:= False;

//Creating and displaying the new array of unique elements

If Unique then

Begin

Arr2[K]:= Arr1[I];

Write(Arr2[K], ' ');

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

WriteLn;

//Displaying the quantity of elements in the new array

If Quantity > 0 then

WriteLn('Quantity of unique elements: ', Quantity)

Else

WriteLn('There is no unique elements in array');

ReadLn;

End.

Приложение Д

(обязательное)

Исходный код программы (по постановке 2)

Массив заполняется путем ввода данных с клавиатуры

Program Lab3Part2Read;

{Find the biggest duplicate of array}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare vars

Var

Arr1, Arr2: Array[1..1000] of Integer;

I, J, K, Quantity, Error, Limit, Max: Integer;

Duplicate: Boolean;

Input: String;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - quantity of duplicates

//Error - operator for checking input

//Limit - number of elements in array

//Max - the biggest duplicate

//Duplicate - if yes, then element has duplicates

//Input - input strings

Begin

//Checking for the correct input

Repeat

Write('Enter the quantity of input array elements: ');

ReadLn(Input);

Val(Input, Limit, Error);

If (Limit <= 0) or (Error <> 0) then

WriteLn('Invalid input. Enter another number.');

Until (Limit > 0) and (Error = 0);

WriteLn;

//Enter every element of array

For I:= 1 to Limit do

Begin

//Checking for the correct input

Repeat

Write('a[', I, ']=');

ReadLn(Input);

Val(Input, Arr1[I], Error);

If Error <> 0 then

WriteLn('Invalid input. Enter another number.');

Until Error = 0;

End;

K:= 1;

//Prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Duplicate:= False;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Duplicate:= True;

//Creating the new array of duplicates

If Duplicate then

Begin

Arr2[K]:= Arr1[I];

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

//Finding the biggest element of the second array

Max:= Arr2[1];

For K:= 2 to Limit do

If Arr2[K] > Max then

Max:= Arr2[K];

WriteLn;

//Displaying the biggest duplicate

If Quantity > 0 then

WriteLn('The biggest duplicate: ', Max)

Else

WriteLn('There is no duplicates in array');

ReadLn;

End.

Приложение Е

(обязательное)

Исходный код программы (по постановке 2)

Массив заполняется с использованием генератора равномерно-распределенных чисел

Program Lab3Part2Generator;

{Find the biggest duplicate of array}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare vars

Var

Arr1, Arr2: Array[1..1000] of Integer;

I, J, K, Quantity, Error, Limit, Max: Integer;

Duplicate: Boolean;

Input: String;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - quantity of duplicates

//Error - operator for checking input

//Limit - number of elements in array

//Max - the biggest duplicate

//Duplicate - if yes, then element has duplicates

//Input - input strings

Begin

//Checking for the correct input

Repeat

Write('Enter the quantity of input array elements:');

ReadLn(Input);

Val(Input, Limit, Error);

If (Limit <= 0) or (Error <> 0) then

WriteLn('Invalid input. Enter another number.');

Until (Limit > 0) and (Error = 0);

WriteLn;

//Generate and display every element of array

Randomize;

//For every I

For I:= 1 to Limit do

Begin

Arr1[I]:=Round(Random(1000)/100 - 5);

Write(Arr1[I], ' ');

End;

WriteLn;

K:= 1;

//To prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Duplicate:= False;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Duplicate:= True;

//Creating the new array of duplicates

If Duplicate then

Begin

Arr2[K]:= Arr1[I];

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

//Finding the biggest element of the second array

Max:= Arr2[1];

For K:= 2 to Limit do

If Arr2[K] > Max then

Max:= Arr2[K];

WriteLn;

//Displaying the biggest duplicate

If Quantity > 0 then

WriteLn('The biggest duplicate: ', Max)

Else

WriteLn('There is no duplicates in array');

ReadLn;

End.

Приложение Ж

(обязательное)

Исходный код программы (по постановке 2)

Массив заполняется с использованием типизированной константы-массив

Program Lab3Part2Const;

{Find the biggest duplicate of array}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare consts

Const

Limit = 10;

Arr1: Array[1..Limit] of Integer = (1, 1, 3, 4, 5, 6, 7, 8, 9, 9);

{Arr1 - array with duplicates

Limit - number of elements in array}

//Declare vars

Var

Arr2: Array[1..Limit] of Integer;

I, J, K, Quantity, Max: Integer;

Duplicate: Boolean;

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - quantity of duplicates

//Max - the biggest duplicate

//Duplicate - if yes, then element has duplicates

Begin

//Display every element of array

For I:= 1 to Limit do

Write(Arr1[I], ' ');

WriteLn;

K:= 1;

//To prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Duplicate:= False;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Duplicate:= True;

//Creating the new array of duplicates

If Duplicate then

Begin

Arr2[K]:= Arr1[I];

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

//Finding the biggest element of the second array

Max:= Arr2[1];

For K:= 2 to Limit do

If Arr2[K] > Max then

Max:= Arr2[K];

WriteLn;

//Displaying the biggest duplicate

If Quantity > 0 then

WriteLn('The biggest duplicate: ', Max)

Else

WriteLn('There is no duplicates in array');

ReadLn;

End.

Приложение З

(обязательное)

Исходный код программы (по постановке 2)

Массив заполняется с использованием типизированной переменной-массив

Program Lab3Part2Var;

{Find the biggest duplicate of array}

//Use app

{$APPTYPE CONSOLE}

//Declare modules

Uses

SysUtils;

//Declare consts

Const

Limit = 10;

//Limit - number of elements in array

//Declare vars

Var

Arr1: Array[1..Limit] of Integer = (1, 1, 3, 4, 5, 6, 7, 8, 9, 9);

Arr2: Array[1..Limit] of Integer;

I, J, K, Quantity, Max: Integer;

Duplicate: Boolean;

//Arr1 - array with duplicates

//Arr2 - array without duplicates

//I, J, K - loop operators

//Quantity - quantity of duplicates

//Max - the biggest duplicate

//Duplicate - if yes, then element has duplicates

Begin

//Display every element of array

For I:= 1 to Limit do

Write(Arr1[I], ' ');

WriteLn;

K:= 1;

//To prevent from random false number

Quantity:= 0;

//For every I

For I:= 1 to Limit do

Begin

Duplicate:= False;

//For every J

For J:= 1 to Limit do

//Condition of duplicate

If (Arr1[I] = Arr1[J]) and (J <> I) then

Duplicate:= True;

//Creating the new array of duplicates

If Duplicate then

Begin

Arr2[K]:= Arr1[I];

//Increment Quantity

Quantity:= Quantity + 1;

//Increment K

K:= K + 1;

End;

End;

//Finding the biggest element of the second array

Max:= Arr2[1];

For K:= 2 to Limit do

If Arr2[K] > Max then

Max:= Arr2[K];

WriteLn;

//Displaying the biggest duplicate

If Quantity > 0 then

WriteLn('The biggest duplicate: ', Max)

Else

WriteLn('There is no duplicates in array');

ReadLn;

End.

Приложение И

(обязательное)

Тестовые наборы

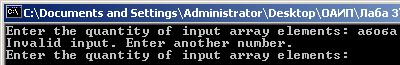
Тестовая ситуация 1: проверка ввода данных (постановки 1, 2)

Тестовая ситуация: проверка, сообщит ли программа о некорректном вводе, если введено буквенное, отрицательное или дробное значение.

Исходные данные: Limit = абоба.

Ожидаемый результат: Invalid input. Enter another number.

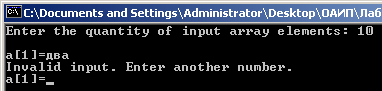
Полученный результат:



Исходные данные: Arr1[1] = два.

Ожидаемый результат: Invalid input. Enter another number.

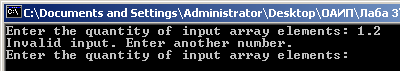
Полученный результат:



Исходные данные: Limit = 1.2.

Ожидаемый результат: Invalid input. Enter another number.

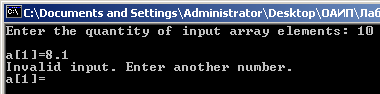
Полученный результат:



Исходные данные: Arr1[1] = 8.1.

Ожидаемый результат: Invalid input. Enter another number.

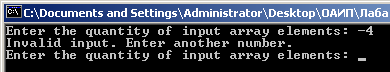
Полученный результат:



Исходные данные: Limit = -2.

Ожидаемый результат: Invalid input. Enter another number.

Полученный результат:



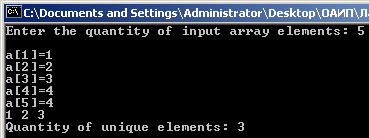
Тестовая ситуация 2: проверка результата (постановка 1)

Тестовая ситуация для проверки вывода правильного результата при нескольких или всех повторяющихся элементах.

Исходные данные: Limit = 5; Arr1[1] = 1; Arr1[2] = 2; Arr1[3] = 3; Arr1[4] = 4; Arr1[5] = 4;

Ожидаемый результат: 1, 2, 3; элементов без повторений - 3.

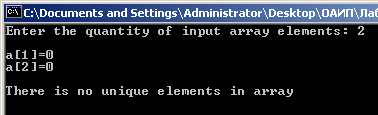
Полученный результат:



Исходные данные: Limit = 2; Arr1[1] = 0; Arr1[2] = 0;

Ожидаемый результат: нет элементов без повторений.

Полученный результат:



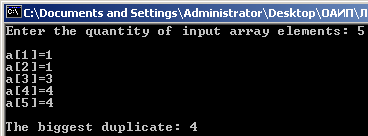
Тестовая ситуация 3: проверка результата (постановка 2)

Тестовая ситуация для проверки вывода правильного результата при нескольких или всех повторяющихся элементах.

Исходные данные: Limit = 5; Arr1[1] = 1; Arr1[1] = 2; Arr1[3] = 3; Arr1[4] = 4; Arr1[5] = 4;

Ожидаемый результат: 4 – крупнейший дубликат.

Полученный результат:



Исходные данные: Limit = 2; Arr1[1] = 0; Arr1[2] = 1;

Ожидаемый результат: нет дубликатов.

Полученный результат:

